IVS Working Group 4: VLBI Data Structures

John Gipson

20th! EVGA Meeting

Bonn, Germany

Working Group Members

Chair	John Gipson
Analysis Coordinator	Axel Nothnagel
Haystack/Correlator Representative	Roger Cappalo
GSFC/Calc/Solve	David Gordon Dan MacMillan
IAA/QUASAR	Sergey Kurbodov Elena Skurhina
JPL/Modest	Chris Jacobs
Occam	Oleg Titov
Vienna	Johannes Boehm
Steelbreeze Formally MAO, now at GSFC	Sergei Bolotin
Observatorie de Paris/PIVEX	Anne-Marie Gontier
NICT	Thomas Hobiger Hiroshi Takiguchi

Working Group Charter

From the IVS website:

The Working Group will ***examine the data structure currently used*** in VLBI data processing and ***investigate what data structure is likely to be needed in the future***.

It will ***design a data structure that meets current and anticipated requirements*** for individual VLBI sessions including a cataloging, archiving and distribution system.

Further, it will ***prepare the transition capability*** through conversion of the current data structure as well as ***cataloging and archiving software*** to the new system.

Current Format

Mk3 database. Currently 30+ years old. Used to archive and transmit IVS sessions.

A product of its time:

- Designed to run on systems with 20k (!!) memory
- Designed using Fortran
- Designed before Fortran had strings

Furthermore…

- Hard to port
- Slow.→ Databases archive information. Superfiles used in analysis.
- Baseline oriented → Tremendous redundancy of some kinds of data.
- Theoretical and observation data mixed.
- Limited user community (20 users?)

Current Format

In spite of its flaws, it has served us well.

- Lasted 30 years—testament to good design.
- Self describing data format.
- Can add new datatypes

Goals

Absolute requirement:

Handle current and anticipated VLBI data needs

Goals

Absolute requirement:

Handle current and anticipated VLBI data needs

Low level goals:

1. Reduce redundancy
2. Ease of access.
3. Speed of access.
4. Different platforms, different languages

Goals

Absolute requirement:

Handle current and anticipated VLBI data needs

Low level goals:

1. Reduce redundancy
2. Ease of access.
3. Speed of access.
4. Different platforms, different languages

High level goals:

1. Flexibility
2. Easy interchange of data.
3. Separation of “observations” from “models” and “theory”
4. Ability to easily access most common parts of the data
5. Ability to access data at different levels of abstraction.
6. Completeness

Goals, Take Two

Goal	Format	Organization	Done?
Low Level Goals			
Reduce Redundancy		✓	
Ease of Access	✓		
Speed of Access	✓		
Many Languages, Platforms	✓		
High Level Goals			
Flexibility		✓	
Easy interchange of sub-sets of the data.	✓	✓	
Separate observables, models, theoreticals		✓	
Separate things that change from things that don't		✓	
Easy access to commonly used parts of the data.		✓	
Data at different levels of abstraction.		✓	
Completeness		✓	

Current MK3 Database

The current Mark3 database is a way of:

1. Storing the data. **Custom Database format.**

Current MK3 Database

The current Mark3 database is a way of:

1. Storing the data.
2. Organizing the data.

Current MK3 Database

Data is organized by Lcodes

1. Type 1 Lcodes. Session data. True for the entire session. (145 items)
   A. Station names, positions.
   B. Source names, positions.
   C. Correlator
   D. π, c, other physical constants?!
   E. …
2. Type 2&3 Lcodes. Observation data. True for an observation. (173 items)
   A. Time
   B. Source
   C. Stations
   D. Station information (Az, El, pressure, calibrations…)
   E. EOP
   F. Observable
   G. Editing
   H. …

Redundancy of Obs-based Data

Many of the observation-dependent data (e.g., Pressure) are really station dependent. This introduces tremendous redundancy.

For an N-Station Scan:
Need to store ***N*** values for Pressure: One for each station.

Database stores by observation.
Number of observations= number of baselines.

N Station scan → N×(N-1)/2 baselines.

For each baseline A-B, stores two values for the pressure: One for station A, one for station B. Total number of values stored:

2× Num baselines = N×(N-1) values.

Data is redundant by a factor of *N-1*

How to Reduce Redundancy

Introduce two new types of data:

1. Station-scan data depends only on the station and the scan.
2. Scan data depends only on the scan.

This requires modest additional bookkeeping:

1. Connects observations to scans.
2. Connects observations to stations.

This can be done, for example using the time-tags of the data.

How to Reduce Redundancy

Introduce two new types of data:

1. Station-scan data depends only on the station and the scan.
2. Scan data depends only on the scan.

This requires modest additional bookkeeping:

1. Connects observations to scans.
2. Connects observations to stations.

This can be done, for example using the time-tags of the data.

With a fair amount of work, we could do this using the *present* Mark3 database format.

Goals

Goal	Format	Organization	Done?
Low Level Goals			
Reduce Redundancy		✓	✓
Ease of Access	✓		
Speed of Access	✓		
Many Languages, Platforms	✓		
High Level Goals			
Flexibility		✓	
Easy interchange of sub-sets of the data.	✓	✓	
Separate observables, models, theoreticals		✓	
Separate things that change from things that don’t		✓	
Easy access to commonly used parts of the data.		✓	
Data at different levels of abstraction.		✓	
Completeness		✓	

Ease of Access

- Ability to easily access data on different platforms.
- Ability to use different languages, platforms.
- Speed

Ease of Access

•Ability to easily access data on different platforms.
•Ability to use different languages, platforms.
•Speed

There are many data storage formats that meet these goals: NetCDF, CDF, HCDF.

Ease of Access

•Ability to easily access data on different platforms.
•Ability to use different languages, platforms.
•Speed

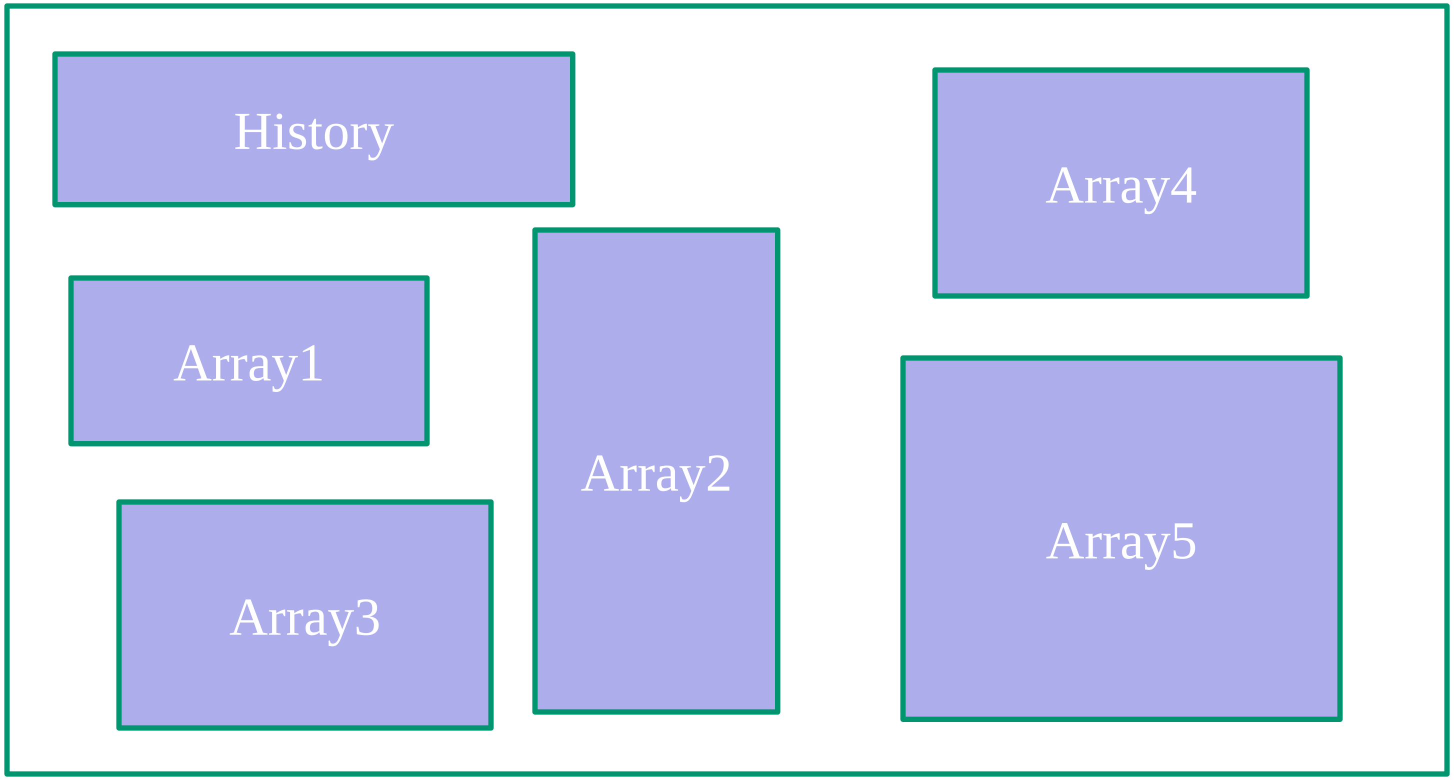
There are many data storage formats that meet these goals: NetCDF, CDF, HCDF.

Recommend using NetCDF4.

1. Wide user community.
2. Many libraries and utilities.
3. Compatible with HCDF.
4. On the fly data-compression.

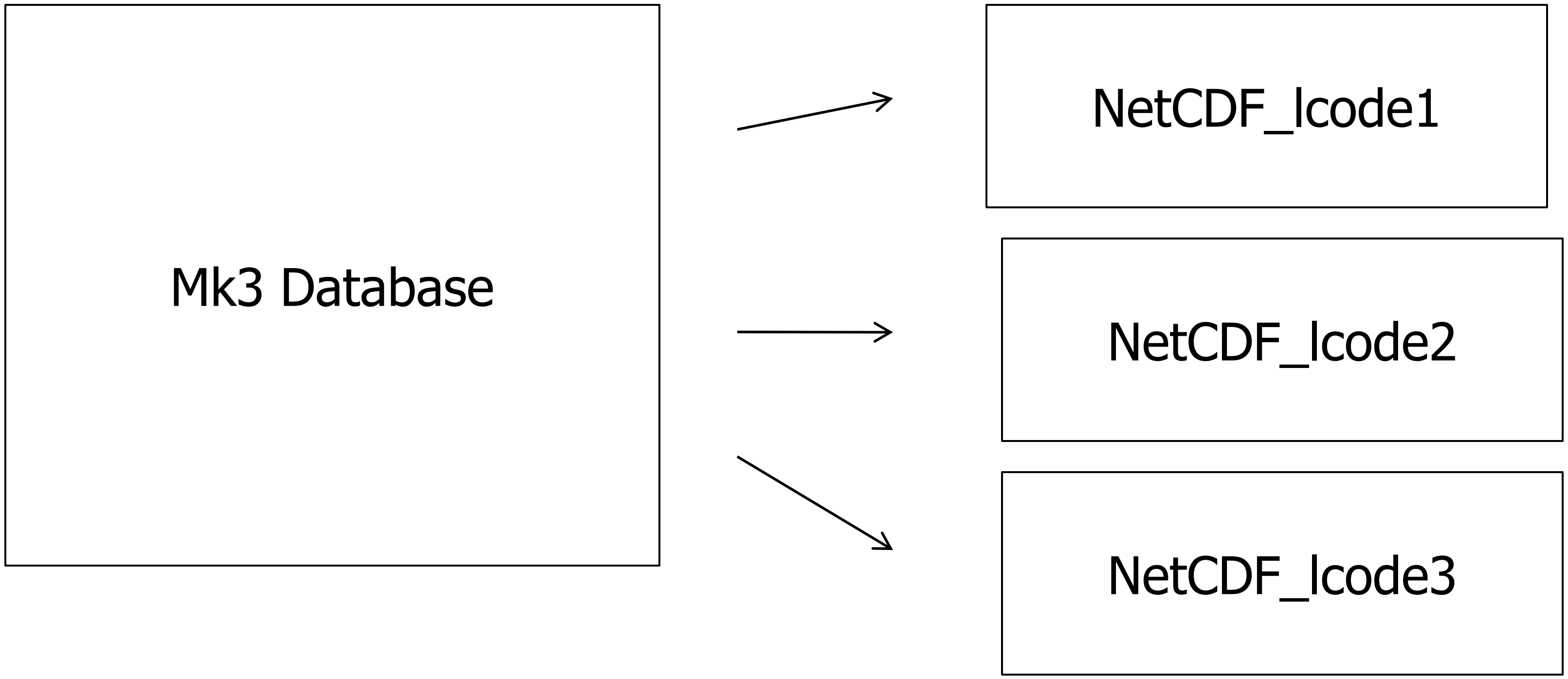
NetCDF4 also makes it possible to access sub-sets of the data.

What does a NetCDF File Look Like?



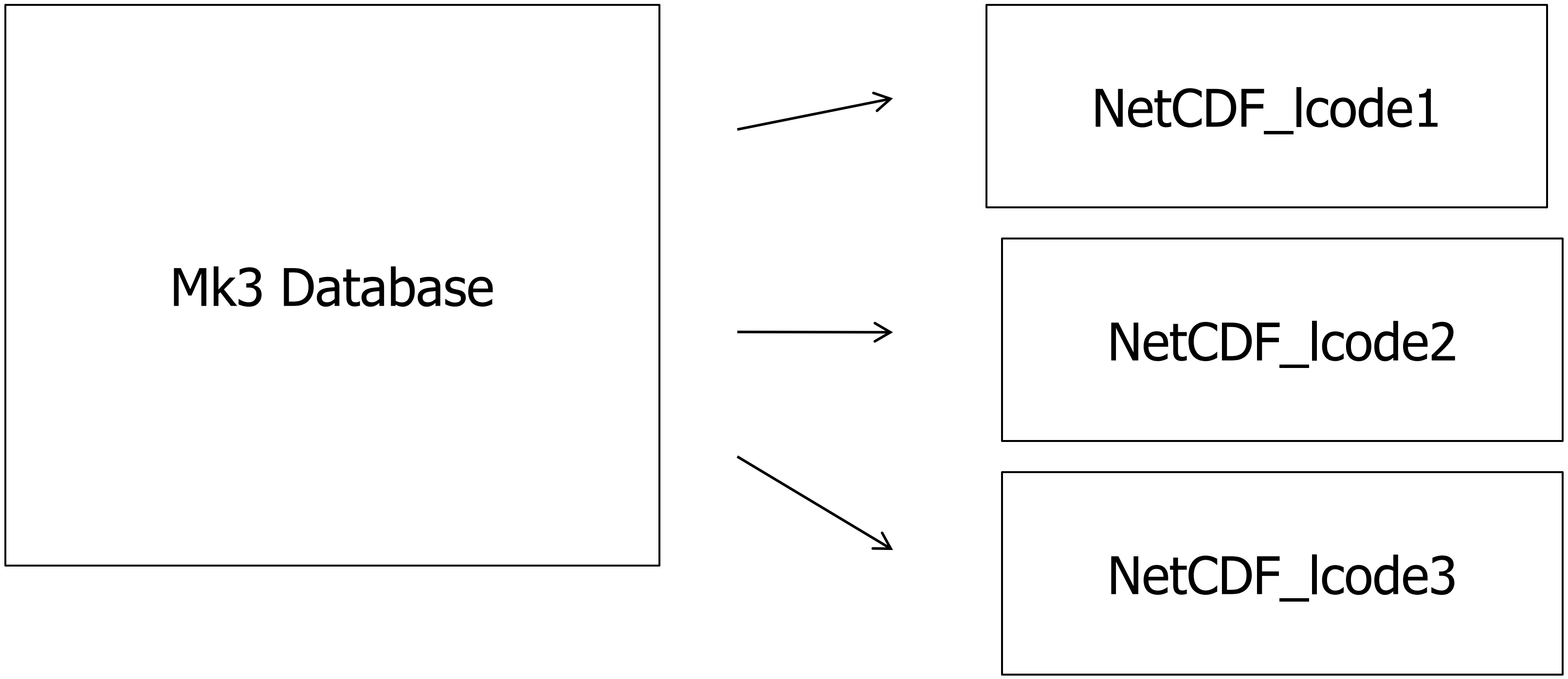
A NetCDF file can contain an arbitrary number of arrays.
The arrays can differ in dimensions and type (byte, short, integer, real, double).
The arrays can have attributes like name, unit, long-name, description associated with them.

Hobiger's NetCDF database implementation



There is a 1-1 mapping between lcodes and NetCDF arrays.

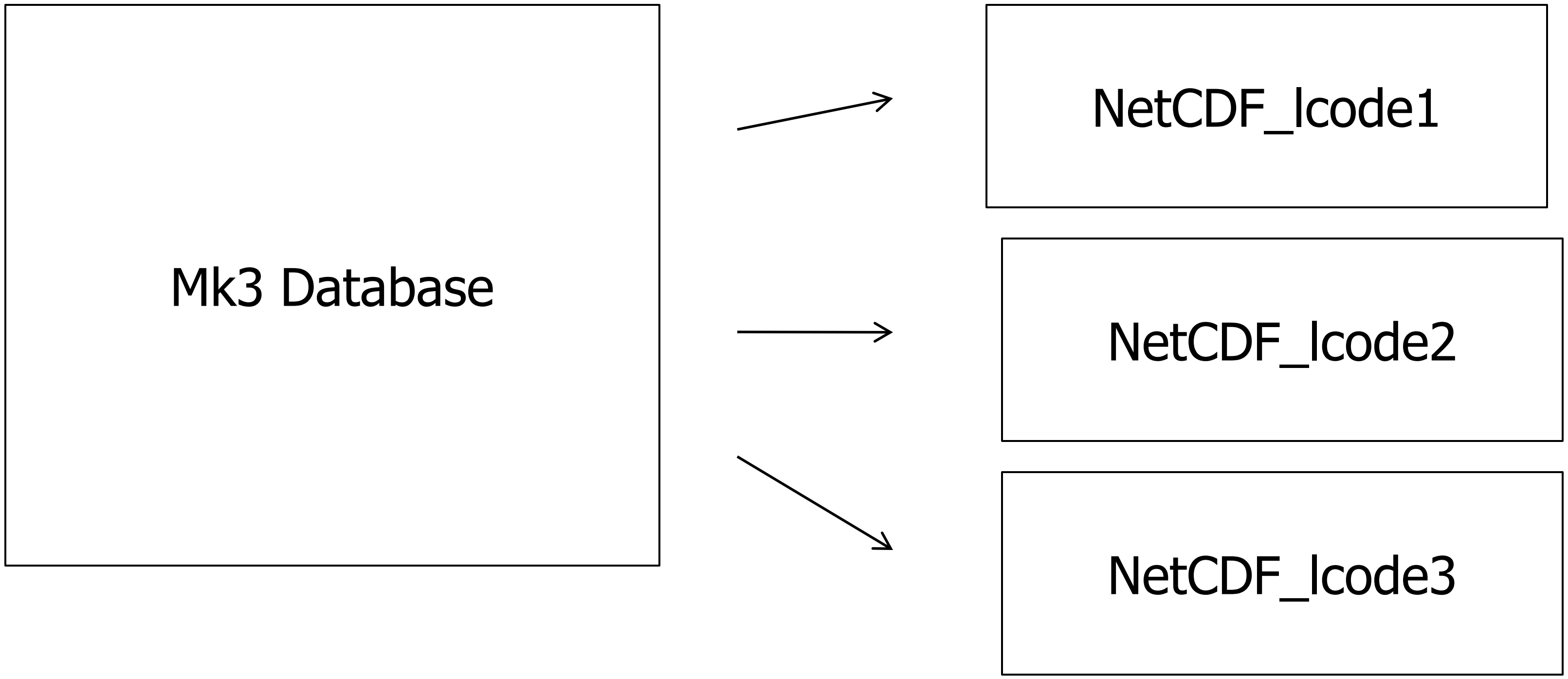
Hobiger’s NetCDF database implementation



There is a 1-1 mapping between lcodes and NetCDF arrays.

Can also go from NetCDF files to Mk3 database.

Hobiger's NetCDF database implementation



There is a 1-1 mapping between lcodes and NetCDF arrays.

Can also go from NetCDF files to Mk3 database.

We could store Mark3 databases in NetCDF format.

Goals

Goal	Format	Organization	Done?
Low Level Goals			✓
Reduce Redundancy		✓	✓
Ease of Access	✓		✓
Speed of Access	✓		✓
Many Languages, Platforms	✓		✓
High Level Goals			
Flexibility		✓	
Easy interchange of sub-sets of the data.	✓	✓	
Separate observables, models, theoreticals		✓	
Separate things that change from things that don't		✓	
Easy access to commonly used parts of the data.		✓	✓
Data at different levels of abstraction.		✓	
Completeness		✓	

Splitting the VLBI Session data

The Mark3 database format was designed so that *all* data pertaining to a session resides in one file.

Advantage: "one-stop-shopping".

Disadvantages:

1. Database contains data of interest only to calc/solve users.
2. Anytime anything changes—calibrations, ambiguities, models—you need a new version of the database.
3. Anytime something is added to the database, you need a new version of the database. Some of the earlier databases are on version 25+.
4. Databases contains lots of obsolete information that is no longer used.

Splitting the VLBI session data

Proposal: Gather data that is similar in

- Scope
- Origin
- Physical effect
- Frequency of change.

Store in its own file.

Organizing the Data With Wrappers

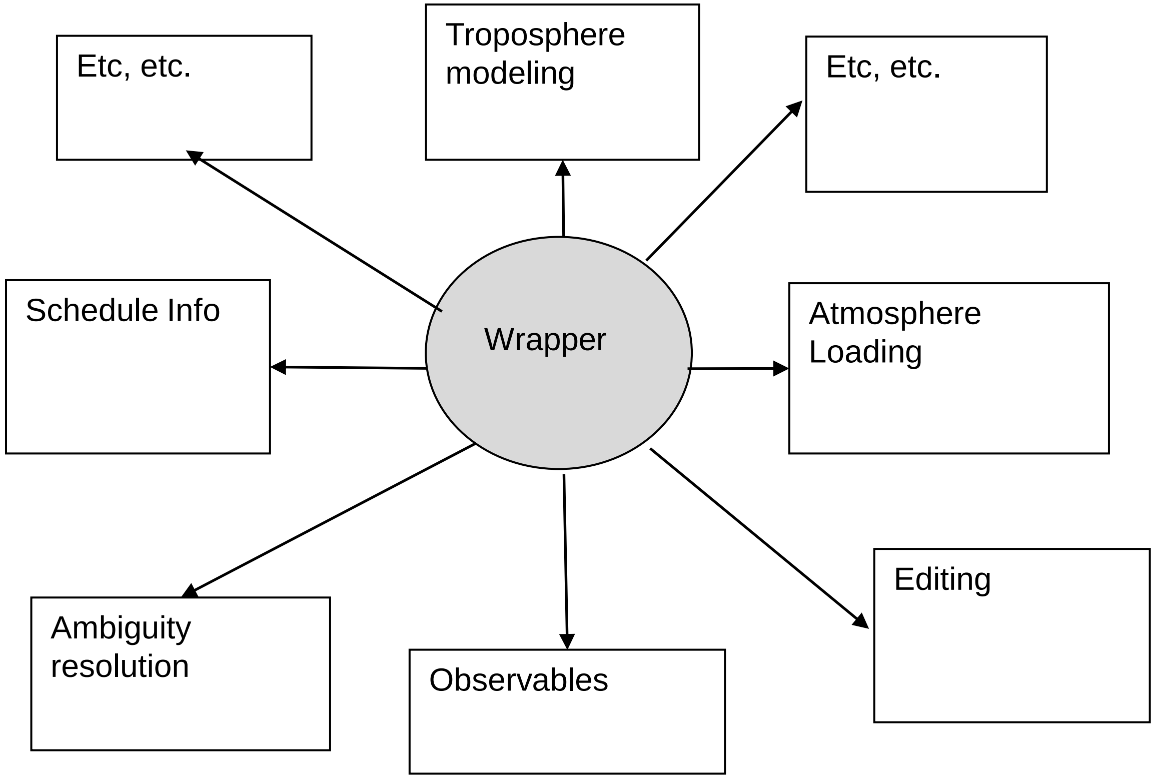
Now that we have split the data, how do we gather it up?

We *wrap* it up using wrappers.

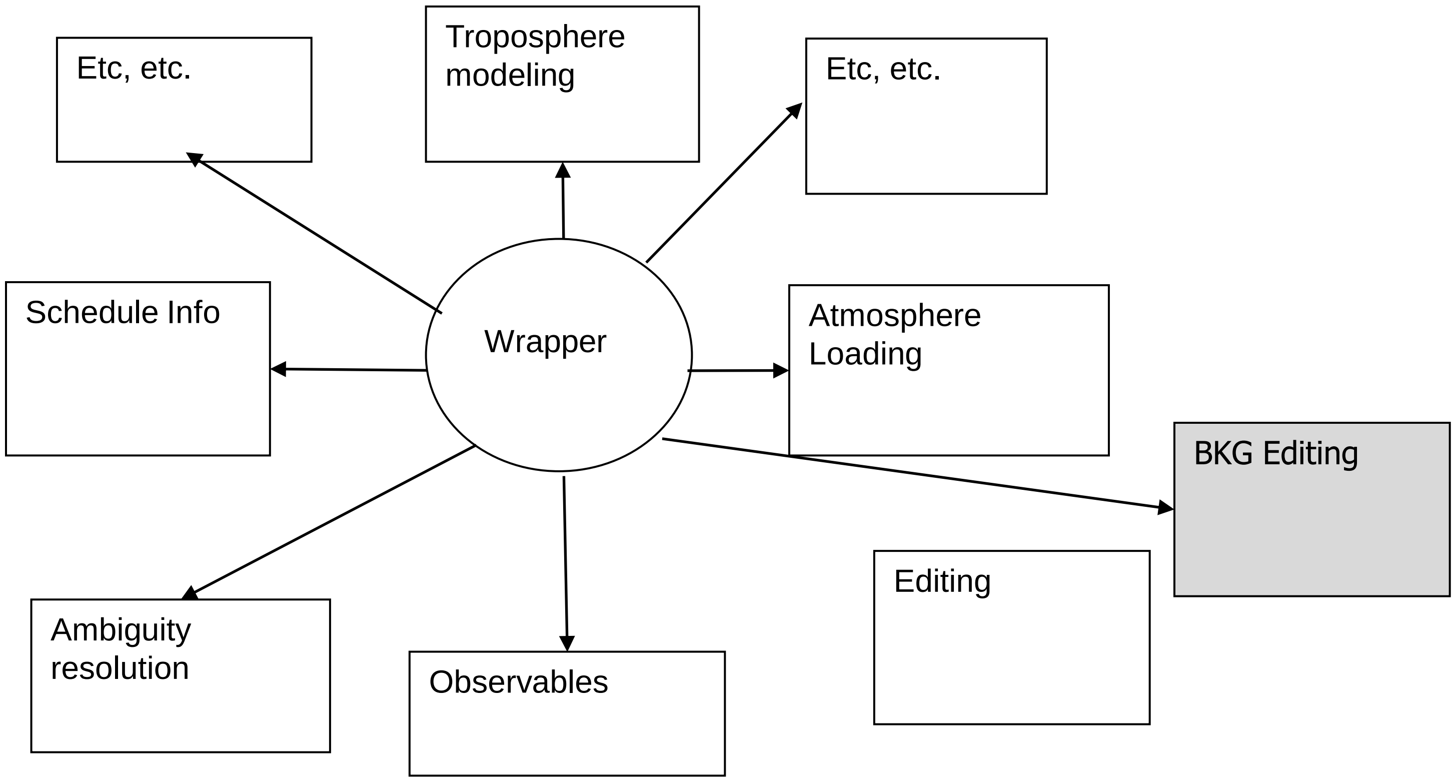
A wrapper is an ASCII file that contains pointers to files that contain data about a session.

A pointer is an instruction about where to find the data. Simplest case is a location on the disk.

Wrappers Organize the Session Data

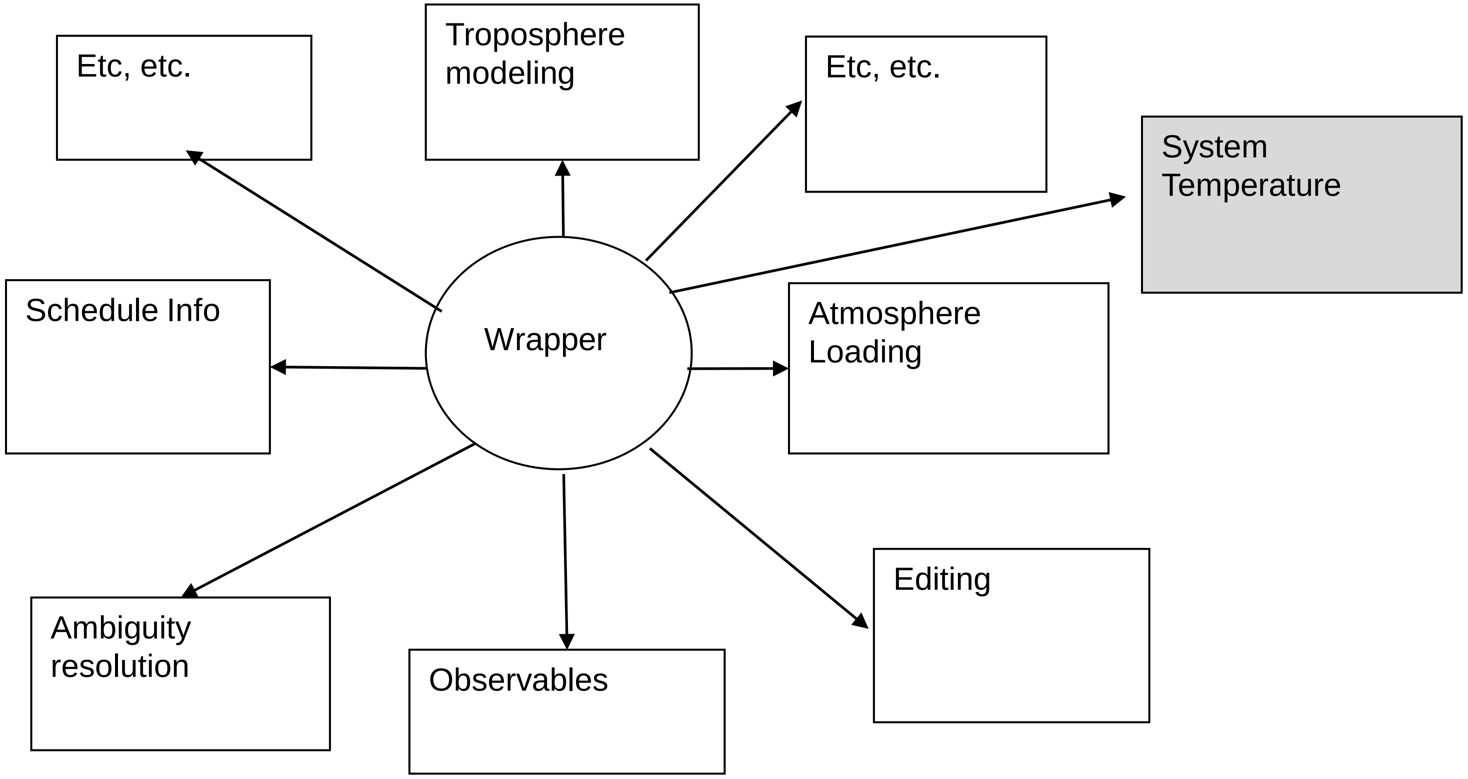


Wrappers Organize the Session Data



Can easily point to different files.

Wrappers Organize the Session Data



Or add new kinds of data

What Does a Wrapper Look Like?

History

Description

Session Data

Kokee Dependent Data

Observation Data

```
Begin History
CreateTimeTag 2009Jun20-12:22:22
Createdby  JohnGipson
End History
! This is a comment.
Begin Description
This is a simple wrapper file for the data in NGS cards.
End Description
! -----another comment.
Begin Session
SessionName I1234
Head.nc
End Session
! ***start the station sections.
Begin Station KOKEE
! KOKEE must be one of the station names in Head.nc
Default_dir KOKEE
AzEl.nc
Met.nc
Cal_kCable.nc
End Station KOKEE
.... OMIT WETTZELL for brevity
! **** Start the observation section
Begin Observation
Default_Dir Obs
ObsIndex.nc
Obs_bX.nc
Obs_bS.nc
Default_Dir ObsEdit
Edit_bX.nc
Ambig_bX.nc
Ambig_bS.nc
Iono_bX.nc
End Observation
```

Goals

Goal	Format	Organization	Done?
Low Level Goals			✓
Reduce Redundancy		✓	✓
Ease of Access	✓		✓
Speed of Access	✓		✓
Many Languages, Platforms	✓		✓
High Level Goals			✓
Flexibility		✓	✓
Easy interchange of sub-sets of the data.	✓	✓	✓
Separate observables, models, theoreticals		✓	✓
Separate things that change from things that don't		✓	✓
Easy access to commonly used parts of the data.		✓	✓
Data at different levels of abstraction.		✓	✓
Completeness		✓	✓

Kinds of Wrappers

1. Bare bones wrapper. Essentially points to information contained in NGS cards.
2. Complete session wrapper. Contains information in NGS cards, also information about where to find correlator output file. Useful for experts.
3. Private wrappers. Used by researchers to test different models.

Where We Are and Next Steps

Draft proposal circulated in July 2009.

1. Positive feedback.
2. Final proposal August, 2011 (after replacement of superfiles).

Conversion of Mark3 database to New Format.

1. Partial utility written in July 2009.
2. Converts ~30% of the lcodes. Everything in NGS cards and more.
3. Current status~95% of the lcodes. Completion “any day now”.

Where We Are and Next Steps

Interface of Software to use OpenDB Format.

1. **Steelbreeze** S. Bolotin
   - A. Partial conversion Sep 2009. Uses NetCDF as storage format.
   - B. Timing penalty of 40 microseconds/obs. No optimization.

     6 million obs * 40 microseconds=240 seconds penalty=6 minutes.
2. **VieVS** . (Vienna VLBI Group). Matthias Madzak.
   - A. Ability to use new format: March-April 2010.
   - B. Introduced new file type—trp.nc. Contains information about troposphere modeling.
3. **Occam**
   - A. Began in fall in 2011
4. C5++: Interface planned.

Calc/Solve Problem

The starting point of most IVS analysis systems is a Version 4 Mark3 database, perhaps converted to NGS cards.

- Ambiguities resolved
- Edited
- Ionosphere

Mark4 databases are made by calc/solve.

- ➢ Calc/solve is *exceedingly* complicated software.
- ➢ Calc/solve and database structure are “joined at the hip”.

Cannot completely abandon databases until calc/solve uses OpenDB format.

Calc/Solve Resolution

Replacement of superfiles by OpenDB format.

A. Began in Fall, 2010. Many of the “lcodes” read from new-format.

B. Complete in Summer 2011.

Replacement of Interactive Solve by νSolve.

A. Creates edited, ambiguity resolved databases.

B. Can read/write Mark3 databases.

C. Can read/write OpenDB format. (not yet implemented).

D. Release later this Summer/Fall.

At this stage could abandon Mark3 databases as exchange format. Still keep as internal calc/solve format to get from correlator output to V3.

Modification/rewrite of software at early steps in the processing chain.

A. Dbedit—makes Version 1 database.

B. Calc—instead of adding lcodes, would write OpenDB

C. Dbcal -- instead of adding clodes, would write OpenDB

Distribution in OpenDB Format

Several Sessions Made Available in July 2009

Intensive, R1 and RDV

Make 1-year available July 1, 2011.

This will give software writers something to work with.

Switch over to new format January 1, 2012

May still need to keep Mark3 databases for internal use by calc/solve to go from correlator output to V3 databases.

Questions?

Questions?